

## **APPENDIX XII - RESOURCE DEPLETION EVENT**

© Copyright 2003 Time Warner Cable, Inc. All rights reserved.

```
import java.lang.*;
5 import java.util.*;

/**
 * Event returned by the system when an uncaught exception or error is encountered.
 */
10 public class ResourceDepletionEvent implements IMessageEvent
{
    public final static int RESOURCE_SYS_MEM_DEPLETED = 0X40000001;
    public final static int RESOURCE_CPU_BANDWIDTH_DEPLETED = 0X40000002;
    public final static int RESOURCE_OUT_OF_SYS_MEM = 0X40000100;
15
    // Static error event Id number. Incremented for each new event.
    private static long lastEventId;

    private long eventId;
20 private long appld;
    private int typeCode;
    private long date;
    private String message;

25 /**
 * Create a SystemErrorEvent.
 *
 * @param errorType - The value indicating the error; see IMessageEvent.
 *
30 * @param errorMsg - The readable error message in the following format:
 */
    public ResourceDepletionEvent(int typeCodeIn, long appldIn, String messageIn)
    {
        lastEventId = (lastEventId == Long.MAX_VALUE) ? 0 : lastEventId + 1;
35        eventId = lastEventId;
        typeCode = typeCodeIn;
        date = System.currentTimeMillis();
        message = messageIn;
    }

40 /**
 * Get the unique identifier of the event.
 *
 * @return unique identifier of the event.
45 */
    public long getEventId()
    {
        return eventId;
    }

50 /**
 * Get the unique identifier of the application that logged the event.
 *
 * @return The application identifier.
55 */
    public long getAppld()
```

```

    {
        return appId;
    }

5  /**
   * Get the event type code. Identifies a code specific to the event system.
   *
   * @return type code of the event.
   */
10 public int getTypeCode()
    {
        return typeCode;
    }

15 /**
   * Set the error type code. Allows an application to change the error received.
   *
   * @param error - The new error value.
   */
20 public void setTypeCode(int typeCodeIn)
    {
        typeCode = typeCodeIn;
    }

25 /**
   * Get the event date in milli-seconds from midnight January 1, 1970.
   *
   * @return Date the event was submitted to the system.
   */
30 public long getDate()
    {
        return date;
    }

35 /**
   * Set the date. Allows an application to change the date received.
   *
   * @param error - The new error value.
   */
40 public void setDate(long dateIn)
    {
        date = dateIn;
    }

45 /**
   * Get the readable message.
   *
   * @return the readable event message.
   */
50 public String getMessage()
    {
        return message;
    }

55 /**
   * Set the error message string. Allows an application to change the error message received.
   *

```

```

    * @param error - The new error message.
    */
    public void setErrorMessage(String messageIn)
    {
5        message = messageIn;
    }

    protected void finalize()
    {
10        // Store the lastErrorEventId in a persistent storage object.
    }

    static {
        if(/*some persistent storage object exists with error event Id stored in it*/ true)
15        lastEventId = /* value in the persistent storage object */1;
        else
            lastEventId = 0;
    }
20 }

```